Antialiasing Complex Global Illumination Effects in Path-space

Laurent Belcour[1], Ling-Qi Yan[2], Ravi Ramamoorthi[3], and Derek Nowrouzezahrai[1,4]
[1]Université de Montréal, [2]UC Berkeley, [3]UC San Diego, [4]McGill University

We present the first method to efficiently predict antialiasing footprints to pre-filter color-, normal-, and displacement-mapped appearance in the context of multi-bounce global illumination. We derive Fourier spectra for radiance and importance functions that allow us to compute spatial-angular filtering footprints at path vertices, for both uni- and bi-directional path construction. We then use these footprints to antialias reflectance modulated by high-resolution maps (such as color and normal maps) encountered along a path. In doing so, we also unify the traditional path-space formulation of light-transport with our frequency-space interpretation of global illumination pre-filtering. Our method is fully compatible with all existing single bounce pre-filtering appearance models, not restricted by path length, and easy to implement atop existing path-space renderers. We illustrate its effectiveness on several radiometrically complex scenarios where previous approaches either completely fail or require orders of magnitude more time to arrive at similarly high-quality results.

Categories and Subject Descriptors: I.3.3 [**Computer Graphics**]: Three-Dimensional Graphics and Realism

General Terms: Global Illumination, Frequency Analysis

Additional Key Words and Phrases: Raytracing, Filtering

1. INTRODUCTION

Texturing with color, normal and displacement maps is a common approach to modelling fine details, increasing a scene's apparent complexity. Whenever such textures are used antialiasing approaches are necessary to avoid objectionable aliasing artifacts. Local prefiltering of color textures [Heckbert 1986] and, most recently, normal and displacement maps [Han et al. 2007; Dupuy et al. 2013; Yan et al. 2014], is a well understood problem; however, all existing antialiasing works only treat *directly visible* surface appearance, projecting the *pixel footprint* onto the geometry. Very little work has investigated the implications of appearance aliasing in the presence of complex light transport with global illumination (GI). Even in this context, techniques such as mipmapping can be applied for incremental benefits, i.e., to reduce the cost of loading full resolution textures into memory.

Prefiltering surface appearance in the presence of GI poses several challenges: since final pixel color results from a multi-dimensional integration of light paths incident on the pixel's footprint (and reflected towards a viewer), we need to express and propagate the *pixel footprint* across multiple bounces (e.g., at each light path vertex; see Figure 5). Intuitively, filtering should be applied at each vertex along a light path, where the final appearance is correctly antialiased by band-limiting its content by the frequency content of the sensor/pixel, as well as emitters/lights and reflectances along the path.

In previous work, ray differentials [Igehy 1999] (Figure 1, left) were devised to predict the pixel footprint from specular interactions, whereas path differentials [Suykens and Willems 2001] are the only technique available to estimate indirect pixel footprints from non-specular interactions (Figure 1, center). Unfortunately, both these approaches only treat eye-paths, leading to incorrect filtering footprints in the common scenario when the light's frequency content is non-negligible. We show how to theoretically account for both the pixel and light's frequency content at a vertex, resulting in the correct antialiasing footprints.

We extend covariance tracing [Belcour et al. 2013] to compute prefiltering bandlimits at path vertices (Figure 1, right), taking into account the sensor *and* emitter spectra, formulating approximate surface footprints from this bandlimit[1]. We also merge two independent unidirectional frequency analyses at path vertex connections, where pixel and light footprints are propagated independently across multiple scene interactions, in order to devise the first bidirectional antialiasing approach. We apply our method to complex GI effects from surfaces with high-resolution normal, color, and displacement maps (Figures 7 to 13). Note that we do not treat the case of volumetric antialiasing that could appear with different objects being clustered in the same filter (geometry antialiasing). Such case cannot be handled by the surface fomulation of light transport we based upon.

Our implementation is straightforward to integrate into modern renderers and we compare our filtered transport algorithms to path-sampling approaches with ray differentials [Igehy 1999] (when available), additionally employing different local appearance prefiltering methods (i.e., Heckbert's diffuse color filtering [1986] and Yan et al.'s specular normal map filtering [2014]). Both traditional uni- and bidirectional algorithms require orders of magnitude longer to converge to the same quality as our approach.

In summary, we present the following technical contributions:

- a frequency domain formulation of *GI prefiltering* (Section 4), including a unidirectional prefiltering analysis,
- an analysis relating traditional path-space transport formulations to their Fourier-domain counterparts, correctly treating spatial-angular sensor *and* emitter spectra in order to devise *bidirectional* vertex antialiasing footprints (Section 5), and
- two efficient techniques for predicting antialiasing footprints and for prefiltering high resolution appearance in GI effects, supporting uni- and bidirectional path tracing (Section 6).

2. PREVIOUS WORK

Prefiltering Local Appearance. Texture prefiltering is a foundational topic in computer graphics, and we refer readers to comprehensive surveys on color texture filtering [Heckbert 1986] and more general local appearance filtering [Bruneton and Neyret 2012]. Several works address *normal map prefiltering* [Fournier 1992; Toksvig 2005] by representing local normal distribution

[1]One would expect that filtering the appearance according to the sampling bandwidth would suffice; however in rendering, appearance is integrated over a surface footprint that can still introduce superfluous high frequencies.

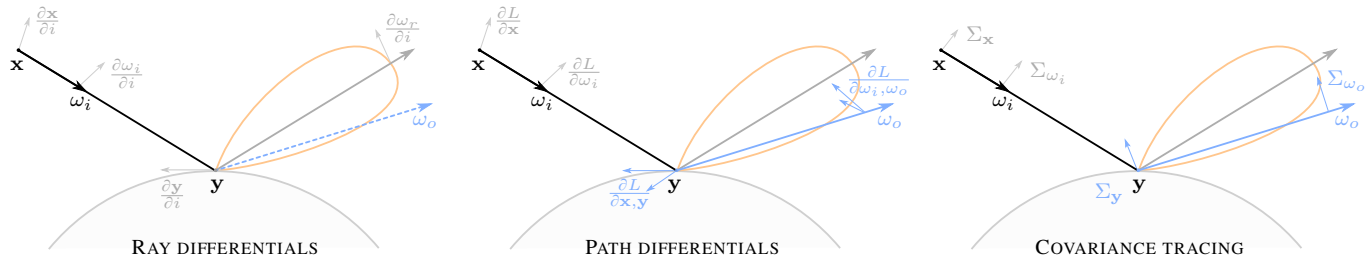


Fig. 1. RAY DIFFERENTIALS (left) propagate the differential of the specular ray interaction with respect to the pixel coordinates. This forbids the study of indirect footprints due to glossy interactions, where a ray spreads across non-mirror directions. PATH DIFFERENTIALS (center) track the gradient of the radiance (or importance), but they require the costly evaluation of partial path derivatives with respect to each position-direction path vertex pair. Our COVARIANCE TRACING approach (right) is simpler, more flexible, and it needs only track the variation of radiance in the local tangent plane of the ray.

functions (NDFs) with bases amenable to fast filtering, often on the GPU [Han et al. 2007; Olano and Baker 2010; Dupuy et al. 2013]. While these interactive techniques are suitable for e.g., video games, none of them correctly treat GI.

Several approaches address more complex micro-structure prefiltering, such as on surfaces with scratches and grooves [Bosch et al. 2004; 2008], and two recent techniques accelerate sampling-based filtering solutions for surfaces with high-resolution NDFs using hierarchical spatial-angular pruning [Yan et al. 2014] and discrete microfacet importance sampling [Jakob et al. 2014]. All of these techniques only antialias the directly-visible appearance of a surface, resorting to an oversmoothed approximate BRDF model as a stopgap solution for indirect bounces.

Prefiltering Indirect Effects. Some works consider special cases of the GI prefiltering problem, often extending the idea of a pixel footprint through to a specific secondary shading effect. Cone tracing [Heckbert and Hanrahan 1984; Shinya et al. 1987] can prefilter geometry and 3D color data [Neyret 1998; Crassin et al. 2009], approximating soft shadows and smooth indirect illumination. Ray differentials [Igehy 1999; Chen and Arvo 2000; Schjøth et al. 2007; Elek et al. 2014] propagate local differentials from sensors (or lights) through only specular reflection and refraction events, but they cannot correctly handle non-specular interactions.

Suykens and Willems' path differentials [2001] most closely resemble our work, albeit with several important differences: while they also identify the importance of devising prefiltering footprints at path vertices, path differentials only account for transport and filtering from eye subpaths and neglect the effects of the light source (and transport from light subpaths), resulting in over-blurring. Moreover, path differentials track and progressively update full path-space partial derivatives, with a compute cost that scales *quadratically* with the *path length* (Figure 1). Our solution avoids this costly computation and scalability issue, maintaining only a single 4×4 symmetric matrix along a path.

Frequency Analysis. We employ a frequency analysis of local light fields [Durand et al. 2005] where, instead of propagating and updating the full 4D spatial-angular light field spectra, we leverage a more compact 2$^{\text{nd}}$-order covariance representation [Belcour et al. 2013; Belcour et al. 2014]. Unlike traditional covariance tracing, we track and update the spectra of radiance *and* importance in order to derive prefiltering footprints along a path. We also derive a simpler BRDF reflection operator for the covariance matrix (Appendix A), and relate our Fourier analysis of path-space filtering to traditional path-space formulations of light transport. This allows us to design uni- and bidirectional path filtering algorithms that are easy to integrate atop existing path-space renderers.

3. OVERVIEW

We will introduce two extended path tracing approaches, each suitable for prefiltering the effects of complex appearance models on the final global illumination. Our **uni-** and **bidirectional antialiasing** approaches (Sections 4 and 5) enable *filtered evaluation* of BSDFs along each path vertex, as illustrated in Figure 2. In Section 6 we detail how our approaches can be readily integrated atop any ray tracer with ray differential support. We present and discuss results in Section 7 before concluding in Section 8.

Unidirectional antialiasing. Starting from the sensor, we propagate a pixel's filter (Figure 2, red) along an eye-path using covariance tracing [Belcour et al. 2013], without considering occlusion. At each scattering interaction, the covariance matrix is first used to compute a spatial footprint which, in turn, is used to perform local filtering of the scattering model at the surface. Here, we employ existing texture filtering [Heckbert 1986] and normal map filtering [Yan et al. 2014] approaches; our technique is insensitive to the choice of approach used to perform the local filtering. After each locally filtered event, the covariance matrix is updated according to an appropriate light transport operation, resulting in an indirect "pixel" filter leaving the surface. Standard path tracing continues, repeating these filtering and evolution steps at each vertex, until path termination.

Bidirectional antialiasing. To antialias appearance in a bidirectional light transport simulation, we simultaneously propagate the covariance matrix representing the pixel's filter for an eye subpath (Figure 2, red) *as well as* the covariance matrix representing the light's importance for the light subpath (Figure 2, blue). As with the unidirectional setting, reflectance profiles at each vertex along both of the subpaths are prefiltered using existing local appearance filtering techniques. The key remaining issue to address in order to correctly filter light transport in a bidirectional setting is the manner in which the eye- and light-subpaths' covariance matrices are combined when the two subpaths are connected: when forming a connection between an eye- and light-subpath, we merge the covariance matrices from each subpath into a new **6D covariance matrix** that captures the combined frequency content of the propagated light and sensor light-fields at a path vertex. We can then use this matrix to compute a filter footprint and antialias the reflectance a vertex without over-blurring the result.

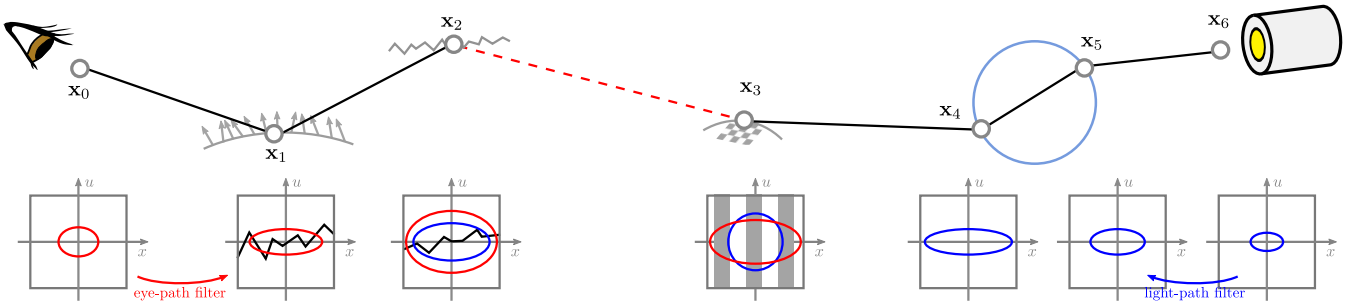


Fig. 2. We compute filtering footprints at path vertices (visualized as 2D spatial-angular distributions, bottom row insets), for both light and eye subpaths, using frequency analysis (Section 4). Given appropriate *local appearance filtering methods*, we can use our footprints to anti-alias high-frequency *global illumination* due to, e.g., high-resolution normal maps ($\mathbf{x}_1$), displacement maps ($\mathbf{x}_2$), or color maps ($\mathbf{x}_3$), after any number of interactions. In bidirectional path tracing ($\mathbf{x}_2 \leftrightarrow \mathbf{x}_3$), eye and light subpath filtering kernels may not match. We derive the correct filtering footprint for such connections in Section 5.

4. PREFILTERING THE RENDERING EQUATION

Our goal is to antialias complex lighting effects, specifically, transport effects that arise from (potentially many) indirect interactions from scenes with realistic reflectance profiles that are modulated by any combination of high-resolution color, normal and displacement maps. Starting with the rendering equation, we derive an expression for the propagation of a spatial-angular footprint (e.g., a pixel footprint) through a scene (Section 4.1). We use frequency analysis (Section 4.2) to derive an easy-to-compute expression for the propagated footprint that we then use to prefilter texture-modulated reflectance models at vertices of the light transport path (Section 4.3).

4.1 Indirect Pixel Footprint

The rendering equation [Kajiya 1986] models global illumination at a shade point as an integral over incoming light directions $\mathcal{S}^2$:

$$L(\mathbf{x},\omega_o) = L^0(\mathbf{x},\omega_o) + \int_{\mathcal{S}^2} L(\mathbf{y}(\mathbf{x},\omega),-\omega)\,\widetilde{\rho}(\mathbf{x},\omega_o,\omega)\,\mathrm{d}\omega, \quad (1)$$

where $L(\mathbf{x},\omega)$ is the radiance at surface point $\mathbf{x}$ (with normal $\mathbf{n}$) in direction ω, $\widetilde{\rho}(\mathbf{x},\omega_o,\omega) = \rho(\mathbf{x},\omega_o,\omega)\max((\mathbf{n}\cdot\omega),0)$ is the spatially-varying cosine-weighted BRDF, $L^0(\mathbf{x},\omega_o)$ is the emitted radiance at $\mathbf{x}$ towards ω_o, and $\mathbf{y}(\mathbf{x},\omega)$ is the closest surface to $\mathbf{x}$ in direction ω. Note that here $\mathcal{S}^2$ is relative to the normal $\mathbf{n}$ at $\mathbf{x}$ but is always an hemisphere.

Equation 1 does not model a pixel's response on the image sensor. This response is defined by a point spread function or, equivalently, a pixel filter function. Given the sensor filter $s^0_{\mathcal{I}}(\mathbf{x},\omega)$ for pixel $\mathcal{I}$, we can extend the rendering equation to one that additionally models the final pixel sensor's response as:

$$L_{\mathcal{I}} = \int_{\Omega_I\times\mathcal{P}_I} s^0_{\mathcal{I}}(\mathbf{x},\omega_o)\,L(\mathbf{x},\omega_o)\,\mathrm{d}\omega_o\,\mathrm{d}\mathbf{x}\,, \quad (2)$$

where $\Omega_I \times \mathcal{P}_I$ is the spatial-angular integration domain of the pixel filter, comprising all directions through the pixel and all image locations over the pixel area. We choose to parameterize the pixel filter using surface points directly visible through the pixel, and with directions pointing towards the pixel at these points, as opposed to points on the sensor and directions leaving the sensor; this choice will simplify our exposition later on.

Substituting the outgoing radiance $L(\mathbf{x},\omega_o)$ in Equation 1 into Equation 2 gives:

$$L_{\mathcal{I}} = L^0_{\mathcal{I}} + \int_{\mathcal{S}^2}\int_{\Omega_I}\int_{\mathcal{P}_I} s^0_{\mathcal{I}}(\mathbf{x},\omega)L(\mathbf{y}(\mathbf{x},\omega_o),-\omega)\,\widetilde{\rho}(\mathbf{x},\omega_o,\omega)\,\mathrm{d}\mathbf{x}\,\mathrm{d}\omega\,\mathrm{d}\omega_o,$$

where $L^0_{\mathcal{I}} = \langle s^0_{\mathcal{I}}, L^0\rangle$ is the directly observed emission at pixel $\mathcal{I}$.

Appearance antialiasing is commonly used in rendering. When computing the *directly-visible* appearance of a surface with e.g., high-frequency variation, many filtering solutions have been proposed: most notably, the product of a pixel filter (projected onto a directly-visible surface) and the spatially-varying BRDF can be pre-integrated, reducing aliasing when shading at low sampling rates. If the incident radiance $L(\mathbf{y},-\omega)$ is almost constant for all surface points directly visible to a pixel, we can approximate the product of the pixel filter and cosine-weighted BRDF with its mean $\overline{\rho}_{\mathcal{I}} = \int s^0_{\mathcal{I}}(\mathbf{x},\omega_o)\widetilde{\rho}(\mathbf{x},\omega_o,\omega)\mathrm{d}\mathbf{x}\,\mathrm{d}\omega_o$. Then, the *antialiased* shading for pixel $\mathcal{I}$ is:

$$L_{\mathcal{I}} \simeq L^0_{\mathcal{I}} + \int_{\mathcal{S}^2}\overline{\rho}_{\mathcal{I}}(\omega)\int_{\mathcal{P}_I} L(\mathbf{y}(\mathbf{x},\omega),-\omega)\,\mathrm{d}\mathbf{x}\,\mathrm{d}\omega\,. \quad (3)$$

Here, we choose to integrate the spatial componnent of L as $\mathbf{y}$ combines $\mathbf{x}$ and ω. Recall also that we restrict the spatial-angular integration domain to the extent of pixel $\mathcal{I}$ (i.e., the domain of $s^0_{\mathcal{I}}$).

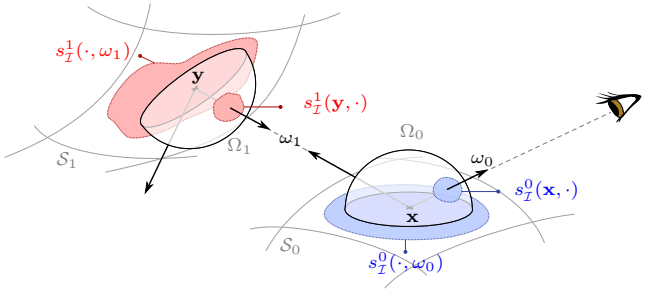


Fig. 3. We recursively construct indirect pixel filters by propagating the interaction of the pixel filter $s^0_{\mathcal{I}}$ with each path vertex's BSDF. Since the indirect pixel filter for bounce-k, $s^k_{\mathcal{I}}$, defines a 4D field, we only visualize 2D slices above: these correspond to angular-distributions (e.g., $s^0_{\mathcal{I}}(\mathbf{x},\cdot)$) when fixing a position (e.g., $\mathbf{x}\in\mathcal{S}_0$), or to spatial-distributions (e.g., $s^1_{\mathcal{I}}(\cdot,\omega_1)$) when fixing a direction (e.g., $\omega_1\in\Omega_1$).

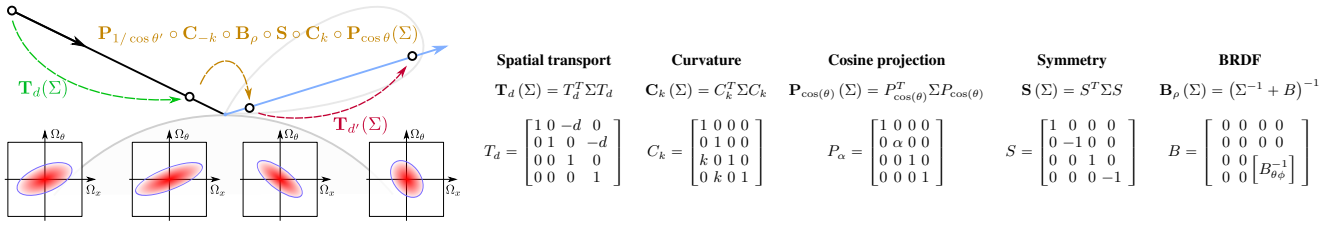


Fig. 4. Covariance tracing [Belcour 2012] predicts the matrix of second moments, Σ, of the light field spectrum (depicted as ellipsoids, bottom left insets). Σ is predicted for a given light path by composing individual matrix operators (top left). These operators have simple analytic forms, as depicted in the right of the Figure: free-space transport and object curvature operators are shears in covariance-space, while the BRDF reflection operator acts as an angular bandlimiting kernel. We introduce a more efficient evaluation of the BRDF matrix operator in Appendix A.

Departing from the *local* filtering scenario of previous work to the *global* transport setting, we observe that the integrated product of the pixel filter and the cosine-weighted BRDF at a directly visible surface point $\mathbf{x}$ can also serve as an *indirect "pixel" filter*, which we can propagate and apply to the incoming radiance at $\mathbf{x}$ (or, equivalently, the outgoing radiance at $\mathbf{y}$; Figure 3). This recursively defines the k-bounce propagated indirect filter from the eye as:

$$s_{\mathcal{I}}^k(\mathbf{y}, \omega) = \int s_{\mathcal{I}}^{k-1}(\mathbf{x}, \omega_o)\, \widetilde{\rho}(\mathbf{x}, \omega_o, \omega)\, \mathrm{d}\omega_o\,, \tag{4}$$

where $\mathbf{y} \in \mathcal{P}_I^k$ is the closest point visible from $\mathbf{x}$ in direction ω. This implicitly defines the propagation of a "filtered" path into the scene.

By combining Equations 1, 2 and 4, we can rewrite the observed (and filtered) pixel intensity as an infinite sum:

$$L_{\mathcal{I}} = L_{\mathcal{I}}^0 + \sum_{k=0}^{\infty} \iint_{\mathcal{S}^2 \times \mathcal{P}_I^k} s_{\mathcal{I}}^k(\mathbf{x}, \omega_o)\, \widetilde{\rho}(\mathbf{x}, \omega_o, \omega)\, L^0(\mathbf{y}(\mathbf{x}, \omega), -\omega)\, \mathrm{d}\mathbf{x}\, \mathrm{d}\omega\,. \tag{5}$$

We use Equation 5 to antialias complex GI effects (after any number of indirect bounces): at each bounce k, we will antialias the cosine-weighted BRDF according to the indirect filter's bandlimit. An alternative interpretation is to imagine the projected "image" of the indirect filter, at a surface along a path (i.e., at a path vertex), as an integration footprint over which we evaluate the mean cosine-weighted BRDF (Figure 5). This view resembles ray differentials, however it is not limited to purely specular interactions. Our rendering method computes the appropriate prefiltering bandlimit, and we will also show how to approximate this footprint in order to support renderers that already use ray differentials (Section 6).

Two scenarios arise when antialiasing multi-bounce shading effects. If we assume that the incident radiance variation at each indirect surface interaction (i.e., each bounce) is negligible compared to the propagated indirect filter's variation, we can simply antialias the local reflectance model at each path vertex's surface according to only the bandlimit of the indirect pixel footprint. This corresponds to a straightforward extension of traditional filtering methods for directly-visible reflectance [Olano and Baker 2010; Dupuy et al. 2013; Yan et al. 2014; Jakob et al. 2014] to multi-bounce effects. We detail this *unidirectional* filtering approach in Sections 4.2 and 4.3, outlining its integration into a unidirectional path tracer.

Whenever the incident radiance's variation cannot be ignored, i.e., when it has a higher frequency content than the (indirect) pixel footprint (e.g., with caustics), we can no longer decouple the filtering integral, and the aforementioned unidirectional scheme will fail to properly antialias the multi-bounce shading effect. To solve this problem, we will show how to evaluate the correct bandlimit, taking the bandlimits of both the propagated indirect pixel footprint *and* the propagated light footprints into account during filtering (Section 5). We will similarly detail the integration of such a filtering scheme atop bidirectional path tracing.

4.2 Unidirectional Connection Covariance Matrix

To filter appearance at arbitrary vertices along a path connecting a pixel to an emitter, we need to be able to estimate the indirect pixel bandwidth at the vertex. We use frequency analysis [Durand et al. 2005] to accurately predict the Fourier spectra of local light fields along a path. Conceptually, this analysis predicts the spatial and angular variation of a light field around any vertex on a light path. We use this theory to consider the evolution of the light field, starting from the sensor's filtering kernel $s_{\mathcal{I}}^0(\mathbf{x}, \omega_o)$.

First, we define the local Fourier Transform (FT) of the indirect pixel filter $s_{\mathcal{I}}^k(\mathbf{x}, \omega)$ (after k bounces), centered around $(\mathbf{x}, \omega)$, as:

$$\widehat{s}_{\mathcal{I}}^k(\boldsymbol{\xi}) = \int_{\mathcal{D}} s_{\mathcal{I}}^k(\mathbf{x} + \mathbf{u}, \omega + \mathbf{v})\, e^{i[\mathbf{u},\mathbf{v}]^T \boldsymbol{\xi}}\, \mathrm{d}\mathbf{u}\, \mathrm{d}\mathbf{v}, \tag{6}$$

where $\boldsymbol{\xi} = [\widehat{\mathbf{x}}, \widehat{\mathbf{v}}]$ is a 1×4 vector of the spatial- and angular-frequency coordinates, $\mathcal{D} = \mathcal{H}^2 \times \mathcal{P}_x$ is the local light field's domain. $\mathcal{H}^2$ is the 2D space of position centered around $\mathbf{x}$ and tangent to ω while $\mathcal{P}_x$ is the 2D space of directions parametrized in

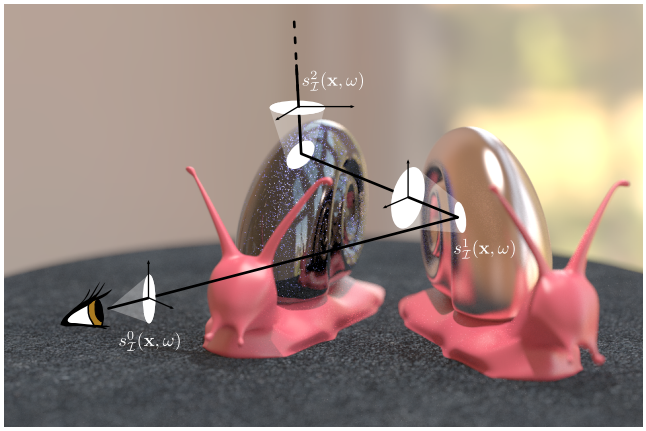


Fig. 5. To antialias complex appearance with global illumination, we propagate pixel footprints along a path through the scene. Expressing the indirect pixel filters $s_{\mathcal{I}}^k$ compactly and accounting for the light interaction is key to correctly and efficiently perform appearance prefiltering.

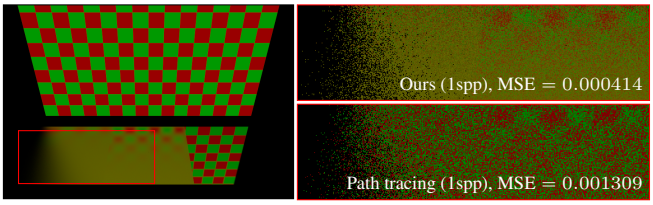


Fig. 6. TWO PLANES scene. We compare our method (top inset) to path tracing with ray differentials (bottom inset) at one sample per pixel. The scene shows the filtering after one indirect bounce by three different BRDFs, from very glossy to mid-roughness, on the ground plane. Our method still exhibits some noise due to random sampling (the BRDF-sampled outgoing direction affects the filter footprint), low sampling rate, and the lack of any geometry/visibility pre-filtering).

tangent space [Durand et al. 2005]. $\widehat{\mathcal{D}} = \widehat{\mathcal{H}^2} \times \widehat{\mathcal{P}_x}$ is the frequency equivalent of $\mathcal{D}$. We denote by $\widehat{g}$ the FT of an arbitrary function g.

Frequency analysis of light transport [Durand et al. 2005] allows us to determine the FT of a spatial-angular distribution propagated along a light path (i.e., $s^k_{\mathcal{I}}(\mathbf{x},\omega)$), using only the FT of the original distribution (i.e., $s^0_{\mathcal{I}}(\mathbf{x},\omega)$) and the FT of the global transport operator $\mathcal{T}$. To do so, we decompose FT of $\mathcal{T}$ into individual transport operations and treat each bounce of transport sequentially:

$$\begin{aligned}\widehat{s}^{k+1}_{\mathcal{I}}(\boldsymbol{\xi}) &= \mathcal{T}\left[\widehat{s}^{k}_{\mathcal{I}}(\boldsymbol{\xi})\right] \\ &= \mathbf{P}_{1/\cos\theta'} \circ \mathbf{C}_{-\kappa} \circ \mathbf{B}_{\rho} \circ \mathbf{S} \circ \mathbf{C}_{\kappa} \circ \mathbf{P}_{\cos\theta} \circ \mathbf{T}_{d}\left[\widehat{s}^{k}_{\mathcal{I}}(\boldsymbol{\xi})\right],\end{aligned}$$

where $\mathbf{P}_{\{\mathrm{x}\}}, \mathbf{C}_{\kappa}, \mathbf{B}_{\rho}, \mathbf{S}$, and $\mathbf{T}_d$ are atomic transport operators used to decompose the global transport operator $\mathcal{T}$ (see Figure 4) and, e.g., applying this compound operation once with $k = 0$ would propagate the sensor filter by one bounce, corresponding conceptually to previous directly-visible appearance filtering work (albeit using our formulation). We refer readers to Chapter 3 of Belcour's thesis [2012] for full derivations of these operators.

To characterize the bandlimit of $\widehat{s}^k_{\mathcal{I}}(\boldsymbol{\xi})$, we use the compact 2nd-order *covariance* representation of Belcour et al. [2013], without considering time nor occlusion. This corresponds to the 4×4 covariance matrix of the 4D FT of the propagated sensor footprint:

$$\Sigma\left(\widehat{s}^k_{\mathcal{I}}\right) = \int_{\widehat{\mathcal{D}}} \left(\boldsymbol{\xi}\cdot\boldsymbol{\xi}^T\right) \widehat{s}^k_{\mathcal{I}}(\boldsymbol{\xi})\,\mathrm{d}\boldsymbol{\xi}\,. \tag{7}$$

The atomic transport operators applied to the 4D light field spectra can all be expressed as matrix operators (applied to the covariance matrix; see Figure 4, right); e.g., the free-space travel operator is:

$$\mathbf{T}_d\left[\,\widehat{s}^0_{\mathcal{I}}(\boldsymbol{\xi})\,\right] \simeq \mathbf{T}_d\left[\Sigma\right] = T_d^T\,\Sigma\,T_d\,.$$

To estimate the covariance matrix after k bounces, Σ_k, we first start at the sensor with covariance Σ_0 and propagate for k bounces using the covariance transport operators, a process illustrated in Figure 4. At each vertex, we can estimate the prefiltering footprint from the covariance matrix to perform antialiasing (Section 4.3).

4.3 Unidirectional Path Filtering with Covariance

We first treat forward path tracing, where paths are constructed from the sensor through the scene and towards emitters. When evaluating transport contributions at path vertices, we can reduce aliasing artifacts from i.e., high-frequency surface's texture, normal, and geometry variations by replacing pointwise evaluations of vertex contribution with *prefiltered values*. This reduces spatial, angular, and temporal noise in the final renderings (see Figures 6 and 7).

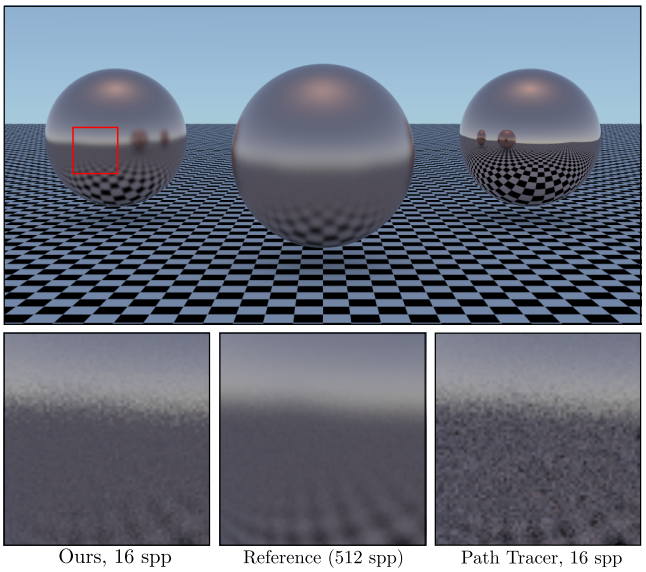


Fig. 7. INDIRECT TEXTURE FILTERING. Our method (bottom left) filters the textured floor after glossy (left & middle spheres) and diffuse (floor) indirect bounces. Even at prohibitively low sampling rates, we outperform path tracing with carefully hand-tuned ray differentials (bottom right).

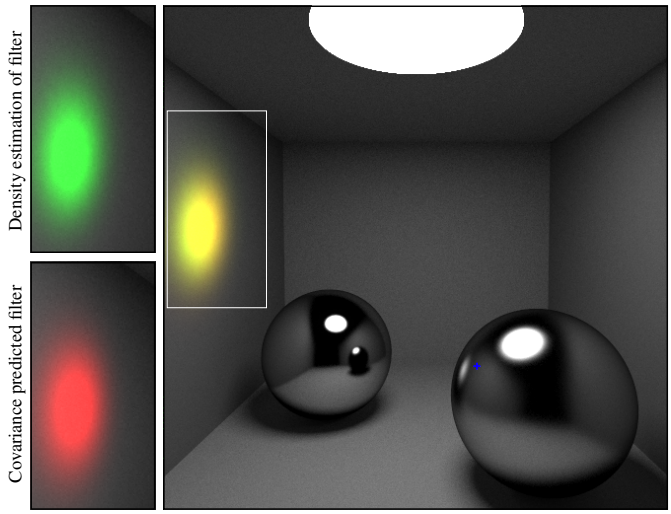


Fig. 8. Our prediction of the spatial filter (in red), after one indirect bounce on the right glossy sphere (see pixel in blue), closely matches a ground truth density estimation of the filter (in green). Specifically, we trace rays in the pixel filter of the blue pixel until a second interaction and perform density estimation (similar to photon mapping) to estimate the ground truth indirect pixel filter.

We filter the spatially-varying BRDF $\rho(\mathbf{x}, \omega_o, \omega)$ at a path vertex $\mathbf{x}_k$, where the effects of any color, normal, and/or displacement map within the indirect pixel filter $s^k_{\mathcal{I}}$ is correctly incorporated into a prefiltered $\widetilde{\rho}$. For example, in the case of high-frequency normal maps, the *effective BRDF* is obtained by integrating the NDF over

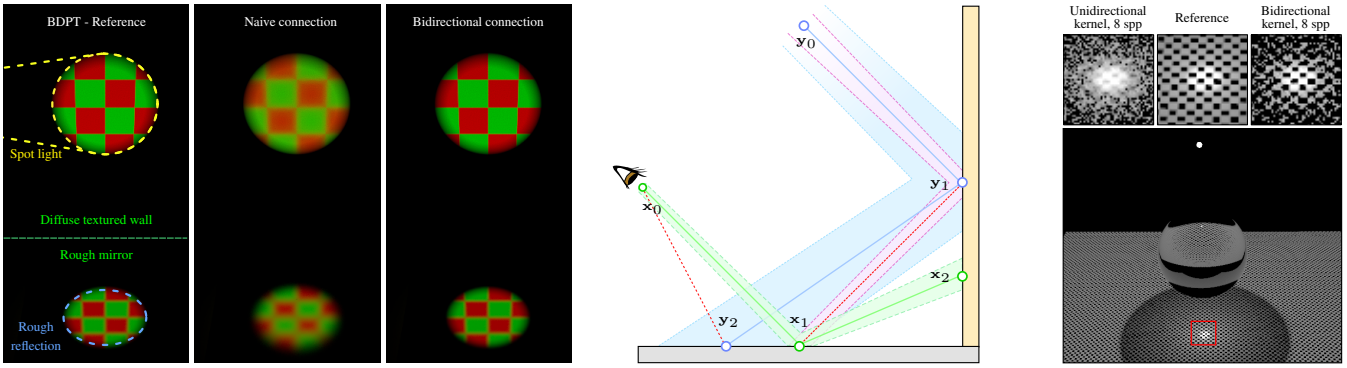


Fig. 9. In BDPT, forming the connection $\mathbf{x}_0 \leftrightarrow \mathbf{y}_2$ (or $\mathbf{x}_1 \leftrightarrow \mathbf{y}_1$) without re-evaluating the antialiased BSDF at vertices $\mathbf{y}_2$ (or $\mathbf{y}_1$ resp.) with a footprint accounting for the eye path frequencies will lead to over-blurring when the eye path frequencies are higher than the light path frequencies. The same argument holds in the reverse configuration. On the right, we compare naïve connections (without re-evaluation) to bidirectionnal connections (with re-evaluation). Since some connections will account for the BSDF antialiased with respect to the light path frequencies, a blur the size of the caustic's footprint will appear (here the caustic is purposefully not fully focused, in order to better illustrate the blurring effect).

the indirect pixel filter's footprint. The filtered appearance is:

$$\widetilde{\rho}(\mathbf{x}_k, \omega_o, \omega) = \int_{\mathcal{F}} \rho(\mathbf{x}_k + \mathbf{u}, \omega_o + \omega, \omega)\, h(\mathbf{u}, \omega)\, \mathrm{d}\mathbf{u}\, \mathrm{d}\omega = \rho * h\,, \quad (8)$$

where $\mathcal{F} = \mathcal{P}_u \times \Omega$ is the 4-dimensional filtering *footprint* (over local positions and directions about $\mathbf{x}_k$) and h is the filtering *kernel*. Once we determine the filtering kernel, we can use existing local filtering methods (see below) to compute Equation 8.

To derive the filtering kernel at $\mathbf{x}_k$, from the propagated sensor covariance Σ_k, we use a Mahalanobis distance of the light field at $\mathbf{x}_k$ to a 0-mean distribution with covariance Σ_k^{-1} as:

$$h(\mathbf{u}, \omega) = g\left([\mathbf{u}, \omega]^T\ \Sigma_k\ [\mathbf{u}, \omega]\right) \Big/ K_g\,, \quad (9)$$

where $g(x)$ is any standard 1D filter and K_g is a normalization term that takes the footprint dimensionality (i.e., 4) into account. For a Gaussian filter $K_g = (2\pi)^2\sqrt{|\Sigma|}$). Note that any normalized 1D filter is compatible with this formulation. We show in Figure 8 that using a Gaussian filter correctly depicts the orientation and spread of the indirect pixel filter after one bounce.

We note, however, that most filtered BRDF models are not defined with respect to a filter g, but instead to a surface footprint $\mathcal{F}$. In these cases, we extract an equivalent footprint from the covariance (Section 6). Note that by using the surface footprint we lose the ability to incorporate the pixel filter's profile during antialiasing.

Figures 7 and 14 illustrate the benefits of our covariance path tracing (CPT) over traditional path tracing with (carefully hand-tuned) ray differentials (PT). Since we account for any BRDF interaction, we are able to prefilter appearance even at indirect bounces. This allows us to efficiently render highly detailed textures using EWA filtering [Heckbert 1986] (Figure 7) and normal maps using the BRDF filtering model of Yan et al. [2014] (Figure 14).

Next, we extend our prefiltering formulation to bidirectional path tracing [Veach and Guibas 1994; Lafortune and Willems 1993].

5. COVARIANCE FILTERING IN PATH SPACE

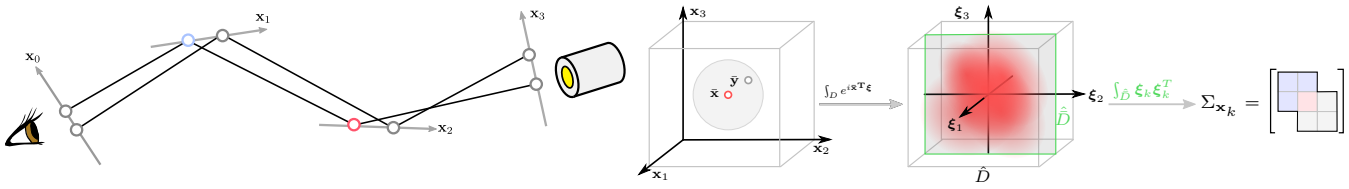
Naïvely extending unidirectional prefiltering to bidirectional path tracing (BDPT)[2] can fail to correctly antialias all vertices along a path (Figure 9). For example, when performing an explicit connection between the light path and the camera (to compute the light image, connection $\mathbf{x}_0 \leftrightarrow \mathbf{y}_2$ in Figure 9), using the BSDF evaluated from the light path's footprint will likely result in overblurring and noticeable artifacts (gray overblur in Figure 9). Since BSDFs are commonly antialiased during path construction and evaluated during connection, we first conclude that **surface appearance must be re-antialiased during connections** in order to avoid these artefacts.

We can similarly extend this idea to account for the eye footprint at the connection vertex; however, this can still lead to over-blurring when the frequency content at vertices closer to the light is much lower in light tracing compared to importance tracing (pink and blue footprints in Figure 9). As such, it is important to devise a theoretically sound expression for the prefiltering kernel of the complete path when a connection is made. We will present such an expression below by building atop the path space formulation of light transport [Veach 1997].

We will show that prefiltering kernels for bidirectional vertex connections can be approximated using the sum of covariance matrices propagated unidirectionally from eye and light sub-paths (Equation 14), permitting simple BSDF antialiasing during subpath connection. Readers not interested by the mathematical derivation of this property can skip to Section 5.4.

We begin by briefly introducing the traditional path space formulation of light transport before performing a local Fourier Transform in this space (Section 5.1). We will derive an expression for the filtering kernel at a given vertex (Section 5.2) and, after applying a Fourier transform to this kernel, we will derive its covariance matrix (relating its computation to the previously defined unidirectional matrices (Section 5.3). Finally, we will discuss how to extend

[2] That is using the footprints derived previously to antialias the eye sub-path and light sub-path appearance independently.



a) Path configuration b) Path space Ω_n c) local Fourier domain $\hat{D}$ d) Vertex covariance matrix

Fig. 10. Consider the 2D case of fixed-length paths interacting with the same surfaces (a), where we window path space (b) in order to perform a local Fourier analysis (c) of the path throughput (we ensure $\mathbf{x}_k$ remains on the same surface). The bandwidth of the throughput's (local) frequency spectrum at a vertex can be compactly represented by the vertex covariance matrix (d). Here, we are interested in the covariance at vertex k (red), with respect to its neighboring vertices $k-1$ and $k+1$ (blue & gray).

BDPT to incorporate appearance antialiasing using the covariance matrices of the filtering kernel (Section 5.4).

5.1 Frequency Analysis of Path Space

Let $\overline{\mathbf{x}} = [\mathbf{x}_0, \ldots, \mathbf{x}_{n-1}]$ be a length-n path, Ω_n the set of all length-n paths (Figure 10 (a) and (b)), and Ω the set of all possible paths. The throughput of a path connecting a sensor pixel to an emitter point is:

$$f(\overline{\mathbf{x}}) = s^0_{\mathcal{I}}(\mathbf{x}_0)\, G(\mathbf{x}_0 \rightarrow \mathbf{x}_1)\, L^0(\mathbf{x}_n) \prod_{k=1}^{n-1} \rho(\mathbf{x}_k)\, G(\mathbf{x}_k \rightarrow \mathbf{x}_{k+1}), \tag{10}$$

where $G(\mathbf{x}_k \rightarrow \mathbf{x}_{k+1})$ accounts for the form factor and visibility between two path vertices, and the remaining terms are path space equivalents of the pixel filter $s^0_{\mathcal{I}}$, BSDF ρ, and light source emission L^0.

The pixel intensity is the accumulation of the integral of this throughput over all path lengths using an appropriate metric (similar to Equation 5):

$$L_{\mathcal{I}} = \sum_{n \in \mathbb{N}} L^n_{\mathcal{I}} = \sum_{n \in \mathbb{N}} \left[\int_{\Omega_n} f(\overline{\mathbf{x}}) \mathbf{d}\mu(\overline{\mathbf{x}}) \right].$$

We define the local Fourier Transform of f around path $\overline{\mathbf{x}}$ as (see Figure 10 (c) and (d)):

$$\hat{f}_{\overline{\mathbf{x}}}(\boldsymbol{\xi}) = \int_{\Omega_n} w([\overline{\mathbf{x}} - \overline{\mathbf{y}}]^T W_{\text{max}} [\overline{\mathbf{x}} - \overline{\mathbf{y}}])\, f(\overline{\mathbf{y}})\, e^{i\overline{\mathbf{y}}^T \boldsymbol{\xi}}\, \mathrm{d}\overline{\mathbf{y}}, \tag{11}$$

where $w([\overline{\mathbf{x}} - \overline{\mathbf{y}}]^T W_{\text{max}} [\overline{\mathbf{x}} - \overline{\mathbf{y}}])$ is a windowing function around $\overline{\mathbf{x}}$ that (conceptually) restricts the analysis to the local subspace where the Fourier Transform is well defined (note that Ω_n is a cartesian product of differential surface manifolds). Specifically, this window ensures that vertices remain on the same surface. It will later be used to ensure that the paraxial approximation of covariance tracing is met. Here, W_{max} is a diagonal matrix that controls the extent of the window, and $\boldsymbol{\xi} = [\boldsymbol{\xi}_0, \ldots, \boldsymbol{\xi}_{n-1}]$ is a $1 \times 2n$ frequency coordinate vector for a length-n path (n *dimensions* with 2 coordinates per dimension; similarly to Equation 6).

In practice, W_{max} will be determined by the renderer's pixel sampling rate, as well as any additional stratification schemes applied during pixel and light sampling (see Section 6 for details).

5.2 Bidirectional Connection Filtering Kernels

Similar to Section 4, we express the contribution of length-n paths to the integrated radiance for a pixel $\mathcal{I}$. However, here we will highlight the appearance at the k^{th} vertex of the path and use the indirect pixel filter and the integrated radiance until vertex $\mathbf{x}_k$:

$$L^n_{\mathcal{I}} = \int s^k_{\mathcal{I}}(\mathbf{x}_{k-1}, \mathbf{x}_k)\, \rho(\mathbf{x}_k)\, L^{n-k}(\mathbf{x}_k, \mathbf{x}_{k+1})\, \mathrm{d}\mathbf{x}_{k-1}\, \mathrm{d}\mathbf{x}_k\, \mathrm{d}\mathbf{x}_{k+1}.$$

Here, $s^k_{\mathcal{I}}(\mathbf{x}_{k-1}, \mathbf{x}_k)$ is the propagated pixel filter and $L^{n-k}(\mathbf{x}_k, \mathbf{x}_{k+1})$ is the integrated incident radiance both defined as:

$$s^k_{\mathcal{I}}(\mathbf{x}_{k-1}, \mathbf{x}_k) = G(\mathbf{x}_{k-1} \rightarrow \mathbf{x}_k) \int s^0_{\mathcal{I}}(\mathbf{x}_0) G(\mathbf{x}_0 \rightarrow \mathbf{x}_1) \times \left[\prod_{i=1}^{k-1} \rho(\mathbf{x}_i)\, G(\mathbf{x}_i \rightarrow \mathbf{x}_{i+1}) \right] \mathrm{d}\mathbf{x}_0 \cdots \mathrm{d}\mathbf{x}_{k-2},$$

and

$$L^{n-k}(\mathbf{x}_k, \mathbf{x}_{k+1}) = G(\mathbf{x}_k \rightarrow \mathbf{x}_{k+1}) \times \int \left[\prod_{i=k+1}^{n-1} \rho(\mathbf{x}_i)\, G(\mathbf{x}_i \rightarrow \mathbf{x}_{i+1}) \right] L^0(\mathbf{x}_n) \mathrm{d}\mathbf{x}_{k+2} \cdots \mathrm{d}\mathbf{x}_n.$$

Note that this is simply a reformulation of $L^n_{\mathcal{I}}$ that highlights the contribution of vertex $\mathbf{x}_k$ to the radiance of length-n paths. Note that we omit the metric in these equations for brevity.

Given the Equations above, and following a similar methodology used when deriving Equation 3, the product of the integrated importance and radiance functions at a vertex can be used as a filtering kernel $h(\overline{\mathbf{x}})$:

$$L^n_{\mathcal{I}} = ||s^k_{\mathcal{I}} \times L^{n-k}|| \int h(\overline{\mathbf{x}}) \rho(\mathbf{x}_k) \mathrm{d}\mathbf{x}_{k-1} \mathrm{d}\mathbf{x}_k \mathrm{d}\mathbf{x}_{k+1},$$
$$\text{with } h(\overline{\mathbf{x}}) = \frac{s^k_{\mathcal{I}}(\mathbf{x}_{k-1}, \mathbf{x}_k) \times L^{n-k}(\mathbf{x}_k, \mathbf{x}_{k+1})}{||s^k_{\mathcal{I}} \times L^{n-k}||}, \tag{12}$$

Where $||f||$ is the norm of f, making $h(\overline{\mathbf{x}})$ normalized.

To determine the filter's bandwidth, we can use its Fourier Transform. The Fourier Transform of $h(\overline{\mathbf{x}})$ is the convolution of the Fourier Transform of each term along the k^{th} component of $\boldsymbol{\xi}$ (noted $*_k$ here):

$$\mathcal{F}[h](\boldsymbol{\xi}) \propto \left[\hat{s}^k_{\mathcal{I}} *_k \hat{L}^{n-k} \right](\boldsymbol{\xi}). \tag{13}$$

5.3 Covariance of Bidirectional Connections

The covariance matrix of the filter defined in Equation 13 can be expressed using the convolution property: *the covariance of the convolution of a zero mean spectra is the sum of the covariances*

of the spectra (see Appendix B):

$$\Sigma_{\mathbf{x}_k} = \Sigma\left(\hat{s}_{\mathcal{I}}^k \ *_k \ \hat{L}^{n-k}\right) = \Sigma\left(\hat{s}_{\mathcal{I}}^k\right) \oplus_k \Sigma\left(\hat{L}^{n-k}\right), \quad (14)$$

where $\Sigma(\hat{s}_{\mathcal{I}}^k)$ and $\Sigma(\hat{L}^{n-k})$ are 4×4 propagated eye- and light-subpath covariance matrices (Section 4.2), and $\oplus_k$ is a restricted matrix sum that only sums the overlapping 2×2 central region of these two covariances. Note, the two 2×2 submatrix blocks in the top-right and bottom-left corners of $\Sigma_{\mathbf{x}_k}$ indicating the correlations between $\mathbf{x}_{k-1}$ and $\mathbf{x}_{k+1}$ cannot be estimated using covariance tracing [Belcour et al. 2013].

Alternatively, we can express the vertex covariance matrix from the Fourier transform of the ratio between the throughput and the reflectance at vertex k, $fr(\overline{\mathbf{x}}) = f(\overline{\mathbf{x}})/\rho(\overline{\mathbf{x}}_k)$, using the Fourier transform defined in Equation 11:

$$\Sigma_{\mathbf{x}_k} = \int_{\widehat{\overline{D}}} (\boldsymbol{\xi}_k \cdot \boldsymbol{\xi}_k^T)\, \widehat{fr}_{\overline{\mathbf{x}}}(\bar{\boldsymbol{\xi}}_k)\, \mathrm{d}\boldsymbol{\xi}_k, \quad (15)$$

where the domain $\widehat{\overline{D}} = \hat{\mathbf{x}}_k \perp \hat{\Omega}_n$ is the restriction of the entire Fourier domain of length-n path space to the three point domain, and $\bar{\boldsymbol{\xi}}_k = [0, ..., 0, \boldsymbol{\xi}_{k-1}, \boldsymbol{\xi}_k, \boldsymbol{\xi}_{k+1}, 0, ..., 0]$ is a $1\times2n$ frequency coordinate vector only capturing the spectrum's covariance for the three point configuration only, averaging out the contribution of the remaining vertices (evaluation at zero in Fourier is equivalent to integrating/averaging in the primal space).

5.4 Using Covariance for Bidirectional Path Filtering

To filter the reflectance $\rho(\overline{\mathbf{x}}_k)$ at a vertex $\mathbf{x}_k$ after a bidirectional connection, we extend the method in Section 4.3: we first evaluate the input eye sub-path and light sub-path covariances until $\mathbf{x}_k$, we compute a filtering footprint using Equation 14, and construct an *equivalent ray differential* (see Section 6, below) from the covariance to evaluate the reflectance.

Standard ray differentials, however, can only be constructed from a 4×4 covariance matrix (where spatial-angular covariance can be converted to spatial-angular differential offsets), and so we need to select one of the two 4×4 submatrix from the 6×6 covariance $\Sigma_{\mathbf{x}_k}$ to construct a ray differential: we can either use the top-left or bottom-right submatrices, corresponding to either the eye or light connection at $\mathbf{x}_k$ (see Figure 10 (d), far right in blue and gray). These options correspond to which "direction" we want to filter from but are equivalent since they incorporate both the eye and light frequency content.

Given our choice of the path vertex covariance submatrix we will use for filtering, the filtering kernel h and footprint $\mathcal{F}$ are exactly the same as in Section 4.3. The antialiased reflectance at the k^{th} vertex can be expressed as the integral of the spatially varying reflectance multiplied by the kernel on the footprint.

6. IMPLEMENTATION DETAILS

We present two new rendering algorithms, covariance path tracing (CPT; based on Section 4) and covariance bidirectional path tracing (CBDPT; based on Section 5), that combine our covariance filtering formulations with covariance tracing to antialias complex global illumination effects. Given the ubiquitous support for ray differentials in modern renderers (i.e., PBRT [Pharr and Humphreys 2010] and Mitsuba [Jakob 2010]), we will derive expressions for our filtering footprints (Sections 4.3 and 5.4) based on *equivalent ray differentials* (see below), allowing us to leverage existing support for efficient texture filtering in modern renderers.

In practice, our algorithms require few changes to existing uni- and bidirectional integrators, described at a high-level as follows:

(1) We compute 4×4 covariance matrices from either the sensor or the light during path tracing: at each interaction, we update the covariance using standard covariance operators (see [Belcour et al. 2013; Belcour et al. 2014]) and store covariance at each path vertex for potential bidirectional connections;
(2) When forming bidirectional connections, we combine the eye and light covariance submatrices to construct the vertex covariance from the eye's direction;
(3) We compute equivalent ray differentials from these footprints (see below) to evaluate texture coordinate partial derivatives with respect to a virtual pixel footprint and to perform the appropriate local texture/appearance filtering.

We now discuss specific implementation details of our algorithms.

Covariance Stratification. In our implementation, pixel sampling rate defines the minimum observable frequency content, corresponding to the diagonal entries of W_{max}. At sensor vertices $\mathbf{x}_0$, we scale the outgoing filter's covariance to conservatively account for any pixel stratification, ensuring that we remain consistent and avoid over-blurring. Note that one could use W_{max} to build a progressive renderer that progressively reduces the bias introduced by antialiasing by increasing the diagonal entries of this matrix.

Bi-directional Covariance Connections. We compute the vertex covariance $\Sigma_{\mathbf{x}_k}$ at a vertex (Section 5.3, Equation 14) using propagated eye and light subpath covariances, $\Sigma(\hat{s}_{\mathcal{I}}^k)$ and $\Sigma(\hat{L}^{n-k})$.

When forming a *filtered bi-directional connection*, we have two choices for the covariance matrix that we use to compute the filtering footprint (Section 5.4): either the eye path submatrix $\Sigma_{\mathbf{x}_{k-1}\to\mathbf{x}_k}$ or the light path submatrix $\Sigma_{\mathbf{x}_k\leftarrow\mathbf{x}_{k+1}}$. In our implementation, we choose $\Sigma_{\mathbf{x}_{k-1}\to\mathbf{x}_k}$ in order to remain compatible with existing appearance antialiasing techniques: all existing models are defined according to an incident 2D surface footprint. This footprint is encoded in the 2D submatrix of vertex k. The remaining components of the covariance matrix describe the angular footprint for antialiasing, which we could use to to more efficiently prefilter appearance in the presence of defocus, for example. We leave this application to future work.

Filtering with Equivalent Ray Differentials. To integrate our work with existing ray-tracing engines, we build atop ray differentials [Igehy 1999]. After each interaction, we build an equivalent ray differential from our covariance matrix to evaluate the prefiltered appearance model.

A ray differential is defined by a central ray $\boldsymbol{r} = \{\boldsymbol{o}, \vec{d}\}$ and its partial derivatives with respect to the pixel position: $\{\partial\boldsymbol{o}/\partial u, \partial\vec{d}/\partial u\}$, $\{\partial\boldsymbol{o}/\partial v, \partial\vec{d}/\partial v\}$. Partial derivatives are taken with respect to the pixel frame $(\vec{u}, \vec{v})$, and the ray differential is propagated into the scene and updated to account for all specular interactions from the sensor.

We can "extract" a ray differential from our 4×4 covariance matrix using a separate eigendecomposition of the spatial and angular submatrices. Assuming a Gaussian filter, we extract the differential

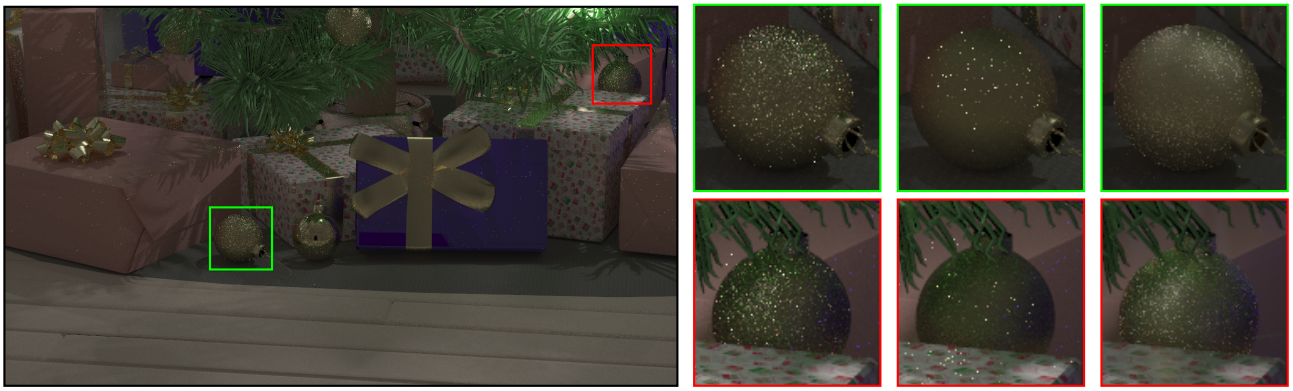


Fig. 11. CHRISTMAS scene. CBDPT renders 1024 spp in 3.78 hours while BDPT can generate 3968 spp in equal time in this complex scene, where emitters are modeled realistically and all visible illumination is from caustics. BDPT completely fails to render the indirect normal mapped microfacet reflections on the ornaments. The reference was run on a cluster of Intel Xeon E5 CPUs and took 1133 core days (**141 days on a 8 core machine**).

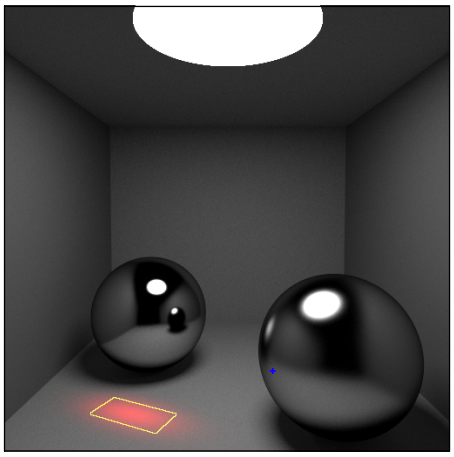


Fig. 12. Our prediction of the spatial filter (in red) after one indirect bounce on the right glossy sphere (see pixel in blue) and the spatial component of the equivalent ray differential (in yellow). This quadrilateral can be used to evaluate the antialiased appearance at the hitpoint location using any nested local appearance filtering method.

using the inverse covariance $C = \Sigma^{-1}$. An eigendecomposition on the first 2×2 block of C gives us the spatial differential:

$$\frac{\partial \boldsymbol{o}}{\partial u} = \frac{\sqrt{\Lambda_{\mathbf{x}_0}}}{2\pi}\mathbf{x}_0 \quad \text{and} \quad \frac{\partial \boldsymbol{o}}{\partial v} = \frac{\sqrt{\Lambda_{\mathbf{x}_1}}}{2\pi}\mathbf{x}_1,$$

where $\{\mathbf{x}_0, \mathbf{x}_1\}$ are the spatial eigenvectors, and $\{\Lambda_{\mathbf{x}_0}, \Lambda_{\mathbf{x}_1}\}$ the spatial eigenvalues. We use a square root of the eigenvalues as they define variance and not standard deviation. The 2π factor is due to the change of domain from the Fourier to the primal. Similarly, we extract the the angular differential using the last 2×2 block of C:

$$\frac{\partial \vec{d}}{\partial u} = \frac{\sqrt{\Lambda_{\boldsymbol{\omega}_0}}}{2\pi}\boldsymbol{\omega}_0 \quad \text{and} \quad \frac{\partial \vec{d}}{\partial v} = \frac{\sqrt{\Lambda_{\boldsymbol{\omega}_0}}}{2\pi}\boldsymbol{\omega}_1,$$

where $\{\boldsymbol{\omega}_0, \boldsymbol{\omega}_1\}$ are the angular eigenvectors, and $\{\Lambda_{\boldsymbol{\omega}_0}, \Lambda_{\boldsymbol{\omega}_1}\}$ the angular eigenvalues. The resulting spatio-angular footprint accurately matches the orientation and spread of the indirect pixel filter (see Figure 12) and allows use to easily integrate our technique into existing rendering engines that are not compatible with anisotropic-Gaussian kernels (e.g. PBRT, Mitsuba). We provide full source code online [Belcour 2016].

7. RESULTS AND DISCUSSION

We implement CPT and CBDPT in Mitsuba [Jakob 2010], with minimal modifications, overloading the RAYDIFFERENTIAL class to perform covariance filter construction (Section 6). We compare our two algorithms to their unfiltered counterparts: forward unidirectional (PT) and bidirectional path tracing (BDPT), both using the reference Mitsuba implementations **with** ray differentials. Our implementation is not optimized and relies as much as possible on existing Mitsuba functionality, with the exception of an (unoptimized) implementation of Yan et al.'s local filtering model [2014] (originally conceived to antialias directly visible high-resolution normal mapped specular surfaces). More recent, high-performance methods (i.e., [Yan et al. 2016]) can be readily substituted into our framework, further increasing our performance; this is particularly noteworthy since our render times are indeed dominated by the evaluation of local appearance filtering models (see our performance breakdown and discussion below). The remaining overhead of covariance tracing and equivalent ray differential computation is negligible.

We demonstrate the usefulness and efficiency of our methods on scenes with complex light transport (TWO PLANES, TEXTURE, SNAILS, ASTRONAUT, CHRISTMAS, and CAUSTIC). All results are computed with multi-threading on a 4-core Intel i7-2600K with 8GB of RAM.

TWO PLANES (Figure 6) illustrates our method's ability to antialias textures with different types of glossy reflections. The bottom plane has three different roughness gradings and reflects the green and red checkerboard. Our method correctly prefilters the reflected appearance of the checkerboard and reduces the image noise significantly, all with *only one sample per pixel*. Note, however, that since we are using BRDF importance sampling with a random number generator, the filtering kernel is not continuous across pixels which explains the residual noise in the result.

TEXTURE (Figure 7) illustrates our method's ability to antialias textures viewed through indirect reflection. We use EWA for albedo filtering and compare our CPT to PT with ray differentials.

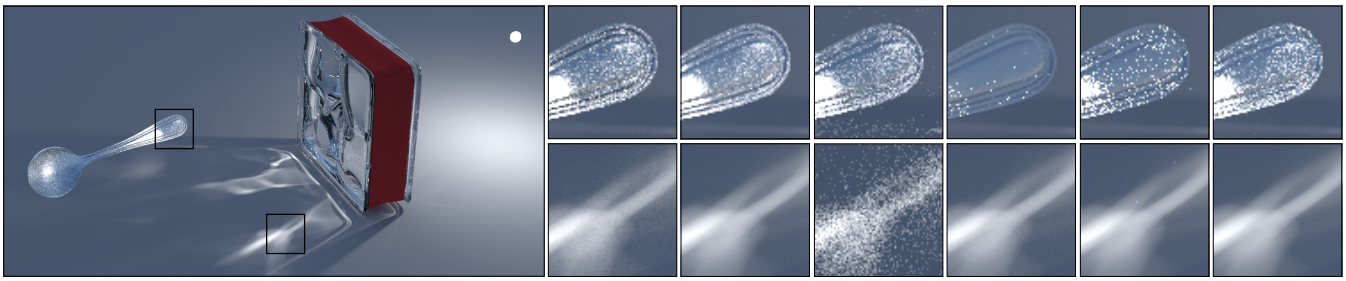


Fig. 13. CAUSTIC scene. We compare our bidirectional method (CBDPT) to PT and BDPT. After 48 minutes, CBDPT begins to visually converge, much faster than both PT and BDPT. BDPT requires prohibitively long render times (here, **24 days**) before converging visually on e.g., the spoon's handle. We report the MSE of the different methods compared to the 1M spp rendering using BDPT (rightmost inset).

We use a low sampling rate to highlight the form of our error versus that of PT. Note that, at this sampling rate, CPT still contains some noise due to integrand discontinuities (i.e., visibility).

SNAILS and ASTRONAUT (Figures 14 and 15) illustrates our ability to correctly predict filtering kernels for indirect bounces, even on challenging cases like the curved snail shell. CPT uses 1.1 mins (1024 spp at 512×512; Figure 14, middle), the majority of which is spent filtering the normal-mapped reflectance [Yan et al. 2014], whereas PT reaches equal error in 18 mins (100K spp; right) and equal time with 5K spp (left). The equal time result has high variance and, despite a roughly equivalent RMSE/SNR, our result's errors are not visually apparent (due to filtering bias). Moreover, the smoothness of our result corresponds much more closely to ground truth and remains temporally coherent (see video). For ASTRONAUT, images are rendered in HD resolution (1280×720). We compare CPT and PT for a roughly equal render time (1024 spp). Despite the relatively simple geometry of the scene, radiometric complexity is high: it is impossible for PT to form a connection between the spoon and the distant star lights after the glossy bounce. We also use BDPT to generate results with high sampling rates (1M and 10M spp), requiring **1 and 10 days** to render *just the inset images*! Using BDPT at these rates to render the full image would have taken **13 and 131 days**. Even at this exagerated setting, the BDPT results have not converged, although they hint shape of the reflection CPT can generate in a negligible fraction of that time.

CHRISTMAS and CAUSTIC (Figures 11 and 13) both use CBDPT. CHRISTMAS is our most complex scene: spherical emitters are embedded in glass bulbs, and so *all emitted light* is due to indirect caustics, which is subsequently reflected indirectly off the glittery Christmas ornaments and shiny presents (both with high-resolution color and normal maps modulating microfacet reflectance models). CAUSTIC casts a complex caustic, resulting from curved reflections and refractions off the displacement mapped glass block, onto the ground and a glittery normal-mapped spoon. CHRISTMAS is rendered at 1280×720 and CAUSTIC at 1024×512. In both cases, BDPT requires a prohibitive amount of time to converge to the correct result (**141 day** render time on 8 cores for 100M spp). Neither PT nor BDPT can resolve *both* the ground caustic and its indirect reflection off the spoon in CAUSTIC.

In the ASTRONAUT and CAUSTIC scenes, "mollifying" the BSDF (as per [Kaplanyan and Dachsbacher 2013]) of the spoon would improve convergence, since the almost-specular interaction could then be replaced with a glossier interaction. Note, however, that the high frequency content of the normal map would then have to be integrated with Monte Carlo integration while our method directly antialiases this effect.

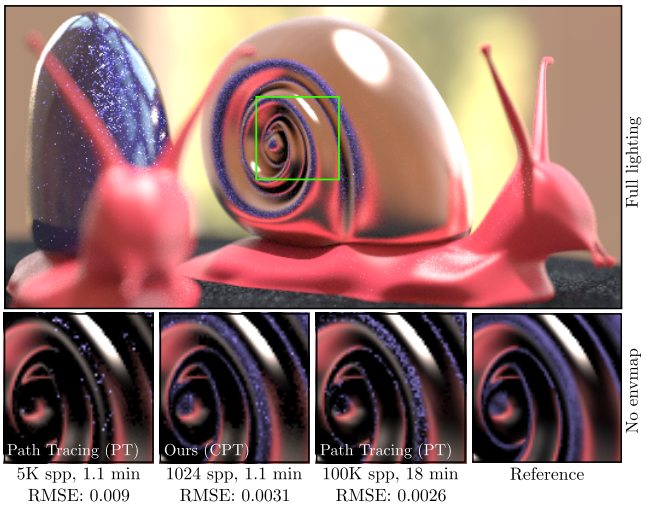


Fig. 14. SNAILS. Equal time & equal error comparison. PT suffers from high-frequency noise (and temporal incoherence; see video) due to aliasing in the indirect reflectance sampling caused by a detailed normal map and environment map. CPT's results visually correspond much more closely to ground truth and remain temporally coherent in animation (see video) despite having roughly equal error (due to filtering bias; see Section 7).

We list performance statistics for covariance tracing and path vertex filtering comparing timings of PT or BDPT with ray differentials ("Alt.") to only tracing covariance ("Cov. only") and to tracing *and* filtering appearance at vertices ("Ours"):

Scene	Alt.	Ours	Cov. only
ASTRONAUT (1024spp)	10.9 min	15.6 min	13.5 min
SNAIL (128spp)	56 s	4 min	1.15 min
CAUSTIC (1024spp)	14.5 min	48 min	19.8 min
CHRISTMAS (1024 spp)	48.35 min	3.8 hours	1.0 hour

From our experiments, covariance tracing without antialiasing adds an overhead of 1.23-$1.36\times$ atop traditional ray tracing (without any fine tuning or optimization of our code). Hence, the majority of our approach's additional cost is due to local appearance filtering (e.g., [Yan et al. 2014]). Note that, at these render times, no existing approach comes close to visual convergence, whereas

all our results do.

Discussion and Future Work. We concentrate on appearance antialiasing on individual objects, leaving the question of geometry antialiasing (i.e., across multiple objects falling within the pixel footprint) to future work. Our method can, however, be used to perform geometry antialiasing when such a pre-filtering model is available to the renderer [Neyret 1998; Crassin et al. 2009]. Note that, in this case, it would be necessary to use the volumetric formulation of covariance tracing [Belcour et al. 2014] for further bounces as the notion of a "surface" becomes ill-defined.

In our implementation, we assumed that the indirect pixel filtering is close to Gaussian shaped (and used Gaussian based antialasing methods). While this is a good approximation for indirect bounces, it isn't true for the first bounce when the pixel filter is not a Gaussian. Since we have a mapping from covariance to ray differentials it is possible to use ray differentials for specular bounces when the pixel filter is a box (note that using other pixel filters shape with ray differentials is incorrect).

Current local appearance antialiasing mostly treat spatial aliasing but largely ignore angular aliasing (some works consider the correlation between view elevation and spatial footprint, however). Filtering according to both spatial and angular footprints could lead to more robust local antialiasing, and our global approach could also readily leverage such extensions, as we already provide the angular bandwidth of indirect pixel/emitter filters as input to the local filtering methods we use.

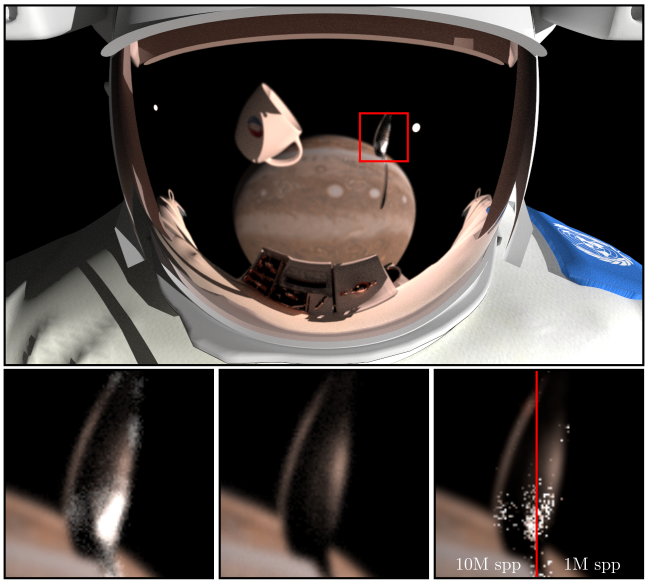
Extensions of our formulation should be able to antialias motion blurred appearance, since covariance tracing inherently supports motion covariance estimation; similarly, since covariance tracing can also be applied to participating media [Belcour et al. 2014], one could generalize our theory to filter volumetric effects so long as local volumetric prefiltering solutions can be devised.

We believe that our method could be combined with filtered importance sampling [Kivánek and Colbert 2007] to form an effective joint-filtering strategy in instances where the number and directions of samples per bounce are fixed directions and known beforehand (such as with low-discrepancy samplers). In these instances, the covariance of the BRDF would be proportional to the square of the sampling *pdf*. We leave this investigation to future work.

Lastly, our bidirectional path space covariance can be applied to problems beyond antialiasing: indeed, it can be used to predict kernel sizes or regularization filters for density estimation and vertex merging algorithms. Specifically, the 2^{nd}-order nature of covariance, and its relation to the Hessian, makes it practical for density estimation [Belcour and Soler 2011; Belcour et al. 2014].

8. CONCLUSION

We have presented an approach to filter complex global illumination appearance, taking the secondary effects of detailed color, normal, and displacement maps into account. We build two complementary theoretical views of the problem, including relating frequency analysis of light transport to path space formulations of the problem, in order to devise unidirectional and bidirectional filtering techniques. Our method is simple to implement in existing renderers and generates equal quality renders for complex images in orders of magnitude less time than the state-of-the-art.



CPT (Ours)
15.6 min (1024 spp)
[**full frame** render time]

PT
15.6 min (1465 spp)
[**full frame** render time]

BDPT
1 & 10 **days** (1M & 10M spp)
[**inset** render time]

Fig. 15. ASTRONAUT scene. We compare our unidirectional CPT to PT and BDPT. PT cannot form the necessary connections to the small star-like luminaires, mandating a filtering approach that supports indirectly-visible objects and high-frequency appearance texturing. Bi-directional approaches are able to form these connections, but they do so with such low probability that multi-day renderings are still unable to visually converge.

Acknowledgements

We would like to thank Pascal Barla and Cyril Soler for their feedbacks during the writing of this work. The Christmas scene was modeled by Jeremy Birn, the Astronaut was modeled by Juan Ignacio Gil-Hutton, the glass brick was modeled by Jose Petri and the Snail model was provided by Yan et al. [2014]. Laurent Belcour and Derek Nowrouzezahrai were supported by NSERC's Discovery and Engage programs. Ling-Qi Yan and Ravi Ramamoorthi were supported by NSF grants 1451828 and 1451830.

REFERENCES

BELCOUR, L. 2012. Frequency analysis of light transport: from theory to implementation. Ph.D. thesis, Grenoble Université.

BELCOUR, L. 2016. Covariance tracing source code. `https://github.com/belcour/CovarianceTracing`. Accessed March-2016.

BELCOUR, L., BALA, K., AND SOLER, C. 2014. A local frequency analysis of light scattering and absorption. ACM Transactions on Graphics 33, 5.

BELCOUR, L. AND SOLER, C. 2011. Frequency-Based Kernel Estimation for Progressive Photon Mapping. In ACM SIGGRAPH Asia Posters. No. 47.

BELCOUR, L., SOLER, C., SUBR, K., HOLZSCHUCH, N., AND DURAND, F. 2013. 5D Covariance tracing for efficient defocus and motion blur. ACM Transactions on Graphics 32, 3, 31:1–31:18.

BOSCH, C., PUEYO, X., MÉRILLOU, S., AND GHAZANFARPOUR, D. 2004. A physically-based model for rendering realistic scratches. In Computer Graphics Forum. Vol. 23. 361–370.

BOSCH, C., PUEYO, X., MÉRILLOU, S., AND GHAZANFARPOUR, D. 2008. A resolution independent approach for the accurate rendering of grooved surfaces. In Computer Graphics Forum. Vol. 27. 1937–1944.

BRUNETON, E. AND NEYRET, F. 2012. A Survey of Non-linear Prefiltering Methods for Efficient and Accurate Surface Shading. IEEE Transactions on Visualization and Computer Graphics 18, 2 (Feb.).

CHEN, M. AND ARVO, J. 2000. Theory and application of specular path perturbation. ACM Transactions on Graphics 19, 1 (Oct.), 246–278.

CRASSIN, C., NEYRET, F., LEFEBVRE, S., AND EISEMANN, E. 2009. GigaVoxels. In ACM Symposium on Interactive 3D Graphics and Games.

DUPUY, J., HEITZ, E., IEHL, J.-C., POULIN, P., NEYRET, F., AND OSTROMOUKHOV, V. 2013. Linear efficient antialiased displacement and reflectance mapping. ACM Transactions on Graphics 32, 6 (Nov.), 1–11.

DURAND, F., HOLZSCHUCH, N., SOLER, C., CHAN, E., AND SILLION, F. X. 2005. A frequency analysis of light transport. ACM Transactions on Graphics 24, 3, 1115–1126.

ELEK, O., BAUSZAT, P., RITSCHEL, T., MAGNOR, M., AND SEIDEL, H.-P. 2014. Progressive Spectral Ray Differentials.

FOURNIER, A. 1992. Filtering normal maps and creating multiple surfaces. Tech. Rep. TR-92-41, Department of Computer Science, University of British Columbia, Vancouver, BC, Canada.

HAN, C., SUN, B., RAMAMOORTHI, R., AND GRINSPUN, E. 2007. Frequency domain normal map filtering. ACM Transactions on Graphics 26, 3, 28.

HECKBERT, P. S. 1986. Survey of texture mapping. Computer Graphics and Applications 6, 11, 56–67.

HECKBERT, P. S. AND HANRAHAN, P. 1984. Beam tracing polygonal objects. ACM SIGGRAPH Computer Graphics 18, 3, 119–127.

IGEHY, H. 1999. Tracing ray differentials. In ACM SIGGRAPH 1999.

JAKOB, W. 2010. Mitsuba renderer. http://www.mitsuba-renderer.org.

JAKOB, W., HAŠAN, M., YAN, L.-Q., LAWRENCE, J., RAMAMOORTHI, R., AND MARSCHNER, S. 2014. Discrete Stochastic Microfacet Models. ACM Transactions on Graphics 33, 4.

KAJIYA, J. T. 1986. The rendering equation. In ACM SIGGRAPH. Vol. 20.

KAPLANYAN, A. S. AND DACHSBACHER, C. 2013. Path space regularization for holistic and robust light transport. Computer Graphics Forum 32, 2, 63–72.

KIVÁNEK, J. AND COLBERT, M. 2007. Real-time Shading with Filtered Importance Sampling. Computer Graphics Forum 27, 4 (jun), 71.

LAFORTUNE, E. P. AND WILLEMS, Y. D. 1993. Bi-directional path tracing. In Proceedings of Compugraphics. 145–153.

NEYRET, F. 1998. Modeling Animating and Rendering Complex Scenes using Volumetric Textures. IEEE Transactions on Visualization and Computer Graphics 4, 1, 55–70.

OLANO, M. AND BAKER, D. 2010. Lean mapping. In ACM I3D. 181–188.

PHARR, M. AND HUMPHREYS, G. 2010. Physically Based Rendering, 2nd Edition. Morgan Kaufmann.

SCHJØTH, L., FRISVAD, J. R., ERLEBEN, K., AND SPORRING, J. 2007. Photon differentials. In GRAPHITE. 179.

SHINYA, M., TAKAHASHI, T., AND NAITO, S. 1987. Principles and applications of pencil tracing. ACM Computer Graphics 21, 4 (Aug.), 45–54.

SUYKENS, F. AND WILLEMS, Y. 2001. Path differentials and applications. In Eurographics Workshop on Rendering. 257–268.

TOKSVIG, M. 2005. Mipmapping normal maps. Journal Graphics Tools 10, 3, 65–71.

VEACH, E. 1997. Robust Monte Carlo Methods for Light Transport Simulation. Ph.D. thesis, Stanford University.

VEACH, E. AND GUIBAS, L. 1994. Bidirectional Estimators for Light Transport. In Proceedings of Eurographics Rendering Workshop.

WOODBURY, M. 1950. Inverting modified matrices. Tech. Rep. 42, Princeton University.

YAN, L.-Q., HAŠAN, M., JAKOB, W., LAWRENCE, J., MARSCHNER, S., AND RAMAMOORTHI, R. 2014. Rendering glints on high-resolution normal-mapped specular surfaces. ACM Transactions on Graphics 33, 4.

YAN, L.-Q., HAŠAN, M., MARSCHNER, S., AND RAMAMOORTHI, R. 2016. Position-normal distributions for efficient rendering of specular microstructure. ACM Transactions on Graphics 35, 4.

APPENDIX

A. IMPROVING THE SCATTERING OPERATOR

We improve the performance of evaluating the band-limiting covariance reflectance operator. The reflectance operator is expressed as a double inversion of the covariance matrix [Belcour et al. 2013, Equation 18]:

$$\Sigma' = \left(\Sigma^{-1} + B^{-1}\right)^{-1},$$

where B is the is the 4×4 rank 2 covariance matrix of the BRDF:

$$B = [0\ 0\ 0\ 0\,;\ 0\ 0\ 0\ 0\,;\ 0\ 0\ b_u\ 0\,;\ 0\ 0\ 0\ b_v]$$

We can avoid computing the costly double matrix inversion with Woodbury's matrix identity [Woodbury 1950]:

$$\Sigma' = \Sigma - \Sigma U\left(B + V\Sigma U\right)^{-1} V\Sigma,$$

with $U^T = V = [0\ 0\ 1\ 0\ 0\,;\ 0\ 0\ 0\ 1\ 0]$, and $B = [b_u\ 0\,;\ 0\ b_v]$. Finally, $B + V\Sigma U$ can be inverted analytically as it is a 2×2 matrix:

$$B + V\Sigma U = \begin{bmatrix} b_u + \sigma_{uu} & \sigma_{uv} \\ \sigma_{vu} & b_v + \sigma_{vv} \end{bmatrix}.$$

B. COVARIANCE MATRIX OF CONVOLUTION

PROPERTY 1. *Belcour's thesis [Belcour 2012, Chapter 4.2] presents a proof that state that the covariance matrix of the convolution of two independent density functions f and g, each with zero mean, is the sum of the covariance matrices of the two functions. Let $\Sigma(f)$ be the covariance of f and $\Sigma(g)$ be the covariance of g, then:*

$$\Sigma(f * g) = \Sigma(f) + \Sigma(g)$$

PROOF. Recall that $\Sigma(f) =_{N\to\infty} \frac{1}{N}\sum_{i=0}^{N} x_i x_i^T$, when $x \sim f$ (if f has zero mean). For this proof, we use the property that a random variable whose *pdf* is a convolution can be expressed as a sum of uncorrelated random variables: $X_{f*g} = Y_f + Z_g$. If we express the unbiased estimator of $\Sigma(f * g)$, we obtain:

$$\Sigma(f * g) =_{N\to\infty} \frac{1}{N}\sum_{i=0}^{N} x_i x_i^T$$

But we can express x_i as a draw of two independent random variables y_i and z_i:

$$\Sigma(f * g) \ = \lim_{N\to\infty} \ \frac{1}{N}\sum_{i=0}^{N}\left[y_i y_i^T + z_i z_i^T + y_i z_i^T + z_i y_i^T\right]$$

The last terms $\sum_{i=0}^{N} y_i z_i^T$ and $\sum_{i=0}^{N} z_i y_i^T$ are the correlation terms and are equal to zero. Thus the estimator of the covariance matrix of the convolution is:

$$\begin{aligned}\Sigma(f * g) \ &=_{N\to\infty} \ \frac{1}{N}\sum_{i=0}^{N} y_i y_i^T + \frac{1}{N}\sum_{i=0}^{N} z_i z_i^T \\ &= \quad \Sigma(f) + \Sigma(g)\end{aligned}$$

□